RECEIVED
CENTRAL FAX CENTER
NOV 17 2006

Amendments to the specification,
Clean version of the replacement paragraph(s)/section(s), pursuant to 37 CFR 1.121(b)(1)(ii):

Please replace the Abstract in its entirety with the rewritten Abstract appended to this Response.

Please replace paragraph [0063] with the following rewritten paragraph:

The current invention also supports several forms of subscriptions. A "subscribed item" specifies an item(s) that a subscriber wants to receive from the primary database. A list of subscribed items can comprise a list of items in the primary database that the subscriber wants to receive. The list can alternatively include a list of items that the subscriber does *not* wish to receive. The following example illustrates these two different subscription forms:

The following amendments place "Program Code" headings on each of the program code sections, pursuant to the Examiner's request that these be explicitly labeled. Please replace paragraph [0074] with the following rewritten paragraph:

Program Code #1:

```
1: {
2:  hash_index = 0
3:  for each character, i.e., i=1, i=2, i=3, and i=4
4:    hash_index += i * char_value(key[i])
5:  hash_index = hash_index modular num_hash_table_slots
6: }
```

Please replace paragraph [0077] with the following rewritten paragraph:

Program Code #2:

```
1: {
2:  hash_index = 0
```

```
3:  for each character
4:    hash_index += char_value(key[i])
5:  hash_index = hash_index modular num_hash_table_slots
6: }
```

Please replace paragraph [0097] with the following rewritten paragraph:

Program Code #3:

```
1: /* NSI: Name Set Index, one NSI per primary site */
2: NSI:
3: {
4:    Function   hash;            /* hash function */
5:    ENTRY      item[MAX_ITEM + 1];  /* hash table */
6:    BM         default;         /* default sites if found no
item in NSI */
7: }
```

Please replace paragraph [0100] with the following rewritten paragraph:

Program Code #4:

```
1: /* Subscribed item, one entry per item */
2: ENTRY:
3: {
4: Char  owner[MAX_LEN_C+1];  /* subscribed item owner */
5: Char  name[MAX_LEN_C+1];   /* subscribed item name */
6: BM    sites;          /* subscribers to this entry */
7: }
```

Please replace paragraph [0101] with the following rewritten paragraph:

(Following line numbers refer to Program Code #4 above.) As shown above at lines 4-5, in the presently preferred embodiment, each individual item is qualified by two literal strings: owner and name. However, a different number of strings could also be used in an

alternative implementation as noted above. In the presently preferred embodiment, the "owner.name" string serves as the hash key for the hash table. Each entry also includes a bitmap string which represents all sites subscribing to a subscribed item, with one bit per subscriber. For instance, if there are three subscribers, the first bit indicates whether the first subscriber is to receive the item, the second bit is an indicator for the second subscriber, and the third bit for the third subscriber.

Please replace paragraph [0103] with the following rewritten paragraph:

Program Code #5:

```
1: /* Bit map string for all subscribers */
2: BM:
3: {
4:  BYTE bitmap[MAX_SITE/8 + 1]   /* one bit per subscriber */
5: }
```

Please replace paragraph [0109] with the following rewritten paragraph:

Program Code #6:

```
1: /* Build a NSI for a given primary site */
2: BUILD(input Site, input/output NSI)
3: {
4:  /* One NSI per primary site */
5:  NSI = empty
6:  For each Sub that subscribes to the primary site Site
7:   ADD(Sub, NSI)
8: }
```

Please replace paragraph [0110] with the following rewritten paragraph:

(Following line numbers refer to Program Code #6 above.) As shown at line 2, the NSI is built for a particular primary site ("input Site"). The Build function starts with an empty name set index as provided at line 5. Given a list of subscribers, each subscriber that subscribes to the primary site (i.e., primary database) is added one at a time. As shown at lines 6-7, the Build function calls the below Add function to add each subscriber one at a time.

Please replace paragraph [0112] with the following rewritten paragraph:

Program Code #7:

```
1: /* Add a subscriber to NSI */
2: ADD(input Sub, input/output NSI)
3: {
4:
5:  Sub_list = subscribed item list of the Sub
6:  Site = the site of the subscriber
7:
8:  /* For a whole set, all items should go to the subscriber*/
9:  /* For a negation set, assume all items go to the subscriber */
10:  If Sub_list is a whole set or a negation set
11:    For each valid NSI.item
12:      NSI.item[Index].sites += Site
13:    NSI.default += Site
14:
15:  /* Add one subscribed item to NSI at a time */
16:  For each "owner.name" in Sub_list
17:    Index = NSI.hash("owner.name")
18:    If NSI.item[Index] does not exist
```

```
19:    /* This is a new item */
20:    NSI.item[Index].owner = "owner"
21:    NSI.item[Index].name = "name"
22:    NSI.item[Index].sites = NSI.default
23:   Else
24:    NSI.item[Index].sites += Site
25:
26:   /* We have assumed that all items go to the subscriber */
27:   /* However, this item should not go to the subscriber */
28:   If Sub_list is a negation set
29:    NSI.item[Index].sites -= Site
30: }
```

Please replace paragraph [0113] with the following rewritten paragraph:

(Following line numbers refer to Program Code #7 above.) The above Add function is called for initially building a name set index (NSI) for a given primary database. It may also be called for adding a subscriber to an existing NSI. When invoked, the Add function adds a subscriber to the NSI. A subscriber may have a list of subscribed items that the subscriber wants to receive. At line 5, "Sub_list" is the list of items that the subscriber wants to receive. This list may comprise a list of the published items that the subscriber is interested in receiving and may include whole sets or negation sets. A whole set may indicate that a subscriber is interested in everything in the primary database. A list may also comprise a negation set indicating that the subscriber is interested in receiving everything from the primary database except for the specific item (or items) that are listed as exclusions. In any case, the Sub_list identifies whatever portion of the primary database that is to be replicated to the subscriber. The "Site" at line 6 identifies the subscriber.

Please replace paragraph [0120] with the following rewritten paragraph:

Program Code #8:

```
1: /* Drop a subscriber from NSI */
2: REMOVE(input Sub, input/output NSI)
3: {
4:     Site = site of the subscriber
5:     NSI.default -= Site
6:
7:     For each valid NSI.item
8:       NSI.item[Index].sites -= Site
9:       If NSI.item[Index].sites = NSI.default
10:        Remove NSI.item[Index]
11: }
```

Please replace paragraph [0124] with the following rewritten paragraph:

Program Code #9:

```
1: /* Search NSI for "owner.name" */
2: SEARCH(input NSI, input owner, input name, output BM)
3: {
4: BM = empty
5:
6: /* Search "owner.name" first */
7: Index = NSI.hash("owner.name")
8: If NSI.item[Index] exists
9:   BM += NSI.item[Index].sites
10:
11: /* If "*.name" is in NSI, its destination subscribes this
"owner.name"*/
```


PAGE 8/26 * RCVD AT 11/17/2006 6:51:56 PM [Eastern Standard Time] * SVR:USPTO-EFXRF-5/10 * DNIS:2738300 * CSID:1 815 572 8299 * DURATION (mm-ss):18-52

```
12: Index = NSI.hash("*.name")
13: If NSI.item[Index] exists
14:   BM += NSI.item[Index].sites
15:
16: /* If "owner.*" is in NSI, its destination subscribes this
 "owner.name" */
17: Index = NSI.hash("owner.*")
18: If NSI.item[Index] exists
19:   BM += NSI.item[Index].sites
20:
21: /* If "owner.name" does not exist in any entry, use default */
22: if BM is empty
23:   BM = NSI.default
24: }
```

Please replace paragraph [0125] with the following rewritten paragraph:

(Following line numbers refer to Program Code #9 above.) When a published item is received by the replication server, the above Search function of the database subscription resolution engine (DSRE) is invoked to resolve which subscribers should receive the published item (i.e., to which subscribers should the published item be replicated). As illustrated above at line 2, the input parameters to the above Search function comprise a name set index (NSI) to be searched, and owner (string), and a name (string). As previously described, the owner and name strings identify a particular item published by the primary database in the currently preferred embodiment of the present invention. The function outputs a bitmap (output BM) indicating the subscribers that should receive the published item.

PAGE 9/26 * RCVD AT 11/17/2006 6:51:56 PM [Eastern Standard Time] * SVR:USPTO-EFXRF-5/10 * DNIS:2738300 * CSID:1 815 572 8299 * DURATION (mm-ss):18-52

Please replace the section heading of the claims (i.e., "Claims" which was automatically generated by the PTO Electronic Stylesheet v. 1.1.1, pursuant to electronic filing), with the following rewritten heading:

What is claimed is:

PAGE 10/26 * RCVD AT 11/17/2006 6:51:56 PM [Eastern Standard Time] * SVR:USPTO-EFXRF-5/10 * DNIS:2738300 * CSID:1 815 572 8299 * DURATION (mm-ss):18-52